

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application Serial No.09/847,534
Filing Date May 1, 2001
Inventorship Novik et al.
Applicant Microsoft Corp.
Group Art Unit 2695
Examiner Bullock Jr.
Attorney's Docket No. MS1-694us
Title: "Method and Apparatus for Correlating Events"

APPEAL BRIEF

To: Commissioner for Patents
PO Box 1450
Alexandria, Virginia 22313-1450

From: Lance Sadler (Tel. 509-324-9256x226; Fax 509-323-8979)
Customer No. 22801

Pursuant to 37 C.F.R. §41.37, Applicant hereby submits an appeal brief for application 09/847,534, filed May 1, 2001, within the requisite time from the date of filing the Notice of Appeal. Accordingly, Applicant appeals to the Board of Patent Appeals and Interferences seeking review of the Examiner's rejections.

<u>Appeal Brief Items</u>	<u>Page</u>
(1) Real Party in Interest	3
(2) Related Appeals and Interferences	3
(3) Status of Claims	3
(4) Status of Amendments	3
(5) Summary of Claimed Subject Matter	4
(6) Grounds of Rejection to be Reviewed on Appeal	8
(7) Argument	8
(8) Appendix of Appealed Claims	31
(9) Evidence appendix	41
(10) Related Proceedings appendix	42

(1) Real Party in Interest

The real party in interest is Microsoft Corporation, the assignee of all right, title and interest in and to the subject invention.

(2) Related Appeals and Interferences

Appellant is not aware of any other appeals, interferences, or judicial proceedings which will directly affect, be directly affected by, or otherwise have a bearing on the Board's decision to this pending appeal.

(3) Status of Claims

Claims 1-25, 27-31, 34-45, and 37-43 stand rejected and are pending in the Application. Claims 1-25, 27-31, 34-45, and 37-43 are appealed. Some of claims 1-25, 27-31, 34-45, and 37-43 were previously amended. Claims 1-25, 27-31, 34-45, and 37-43 are set forth in the Appendix of Appealed Claims on page 31.

(4) Status of Amendments

A first Office Action was issued on December 31, 2003.

A Response was filed on March 31, 2004. Claims 1, 11, 28, and 35 were amended. Claims 40-43 were added.

A Final Office Action was issued on June 15, 2004.

A Response to the Final Office Action was filed July 20, 2004. No amendments were made as part of this Response.

An Advisory Action was issued on August 30, 2004, indicating that the request for reconsideration had been considered but did not place the application in condition for allowance.

Appellant filed a Notice of Appeal on September 24, 2004 in response to the Advisory Action and the Final Office Action.

An Appeal Brief was filed on December 23, 2004.

In view of the Appeal Brief, Prosecution was reopened and an Office Action was filed on April 6, 2005 setting forth a new ground of rejection.

A Response was filed on October 6, 2005. Claims 1, 2, 9, 11-13, 16, 20, 28, 35, 37 and 40-41 were amended. Claims 26, 32-33, and 36 were canceled without prejudice.

A Final Office Action was filed on December 23, 2005.

A Response to the Final Office Action was filed on February 22, 2006.

An Advisory Action was issued on March 24, 2006, indicating that the request for reconsideration had been considered but did not place the application in condition for allowance.

Appellant filed a Notice of Appeal on May 23, 2006 in response to the Advisory Action and the Final Office Action.

(5) Summary of Claimed Subject Matter

A concise explanation of each of the independent claims is included in this Summary section, including specific reference characters. These specific reference characters are examples of particular elements of the drawings for

certain embodiments of the claimed subject matter and the claims are not limited to solely the elements corresponding to these reference characters.

With regard to claim 1, a computer-implemented method comprises: receiving a plurality of events (page 12, lines 3-10; page 15, lines 10-22; Fig. 3, 302; Fig. 6, 602); applying the plurality of events to a correlation function, wherein the correlation function is implemented as a state machine and is configured to correlate the plurality of events (page 14, line 12 – page 15 line 3; page 15 line 23 – page 16 line 6; Fig. 5, 508); identifying an event to which an update consumer has subscribed, wherein the update consumer is associated with the state machine (page 15 line 23 – page 16 line 6; Fig. 7, 706); applying the update consumer to the state machine in response to the identified event (page 15 line 23 – page 16 line 6; Fig. 7, 708); and generating a specific event if the correlation function is satisfied by the plurality of events (page 15, lines 10-22; Fig. 6, 606, 610).

With regard to claim 11, a computer-implemented method comprises: receiving a plurality of events (page 12, lines 3-10; page 15, lines 10-22; Fig. 3, 302; Fig. 6, 602); receiving a plurality of data elements (page 15, lines 10-22; Fig. 6, 602); identifying a plurality of correlation functions configured to correlate the plurality of events and the plurality of data elements, wherein each correlation function is implemented with an associated state machine, and wherein each state machine has an associated update consumer that updates the state of the associated state machine (page 15 lines 10 – page 16 line 6; Fig. 6, 602, 604; Fig. 7, 708); applying the plurality of events and the plurality of data elements to the plurality of correlation functions (page 15, lines 10-22; Fig. 6, 604); and generating a

specific event if at least one of the plurality of correlation functions is satisfied (page 15, lines 10-22; Fig. 6, 610).

With regard to claim 20, a computer-implemented method comprises: identifying a schema for creating state machines (page15 line 23 – page 16 line 6; Fig. 7, 702); the state machines to correlate at least two events (page15 line 23 – page 16 line 6; Fig. 7); creating an instance of a particular state machine (page15 line 23 – page 16 line 6; Fig. 7, 704); defining transitions for the particular state machine by subscribing to at least one event (page15 line 23 – page 16 line 6; Fig. 7, 706); and applying an update consumer to the particular state machine to update the state of the particular state machine, wherein the update consumer is a class object (page15 line 23 – page 16 line 6; Fig. 7, 708).

With regard to claim 28, an apparatus comprises: a plurality of event consumers (page 12 line 3 – page 13 line 12; Fig. 3, 316, 322, 328, 334); and an event correlator coupled to the plurality of event consumers (page 12 line 3 – page 13 line 12; Fig. 3, 310), the event correlator to receive events from at least one event source and to receive data elements from at least one data source, the event correlator further to receive at least one correlation function configured to correlate events and data elements and to apply the received events and the received data elements to the correlation function (page 14, line 12 – page 15 line 3; page 15 line 23 – page 16 line 6; Fig. 5, 508), wherein the correlation function is implemented by a state machine having an associated update consumer that updates the state of the state machine (page15 line 23 – page 16 line 6; Fig. 7, 708), and wherein the event correlator generates a specific event if the received

events and the received data satisfy the correlation function (page 15, lines 10-22; Fig. 6, 610).

With regard to claim 35, one or more computer-readable media having stored thereon a computer program that, when executed by one or more processors, causes the one or more processors to: receive a plurality of events (page 12, lines 3-10; page 15, lines 10-22; Fig. 3, 302; Fig. 6, 602); identify a plurality of correlation functions configured to correlate the plurality of events, wherein each of the plurality of correlation functions is implemented as a state machine having an associated update consumer (page 15 lines 10 – page 16 line 6; Fig. 6, 602, 604; Fig. 7, 708); apply the plurality of events to the plurality of correlation functions to determine whether any of the plurality of correlation functions are satisfied by the plurality of events, wherein the plurality of events are applied by causing update consumers associated with each event to update the state of the associated state machine (page 15 line 23 – page 16 line 6; Fig. 7, 708); and generate a specific event if one of the plurality of correlation functions is satisfied by the plurality of events (page 15, lines 10-22; Fig. 6, 610).

With regard to claim 40, a computer-implemented method comprises: receiving events from event providers (page 12, lines 3-10; page 15, lines 10-22; Fig. 3, 302; Fig. 6, 602); creating a first state machine (page 15 line 23 – page 16 line 6; Fig. 7, 704); creating a second state machine (page 15 line 23 – page 16 line 6; Fig. 7, 704); associating a first event type with the first state machine, wherein the first state machine has an associated first update consumer to update the state of the first state machine (page 15 line 23 – page 16 line 6; Fig. 7, 706); associating a second event type with the second state machine, wherein the second

state machine has an associated second update consumer to update the state of the second state machine (page 15 line 23 – page 16 line 6; Fig. 7, 706); in response to receiving an event having a first event type, applying the first update consumer to the first state machine (page 15 line 23 – page 16 line 6; Fig. 7, 708); in response to receiving an event having a second event type, applying the second update consumer to the second state machine (page 15 line 23 – page 16 line 6; Fig. 7, 708); and if the events are correlated: generating an additional event; and sending the additional event to an event consumer (page 15, lines 10-22; Fig. 6, 610).

(6) Grounds of Rejection to be Reviewed on Appeal

Claims 1-25, 27-31, 34, 35 and 37-43 stand rejected under 35 U.S.C. § 103(a) as being obvious over U.S. Patent No. 6,336,139 to Feridun et al. (hereinafter “Feridun”) in view of U.S. Patent No. 6,751,753 to Nguyen et al. (hereinafter “Nguyen”).

(7) Argument

- A. The rejections under 35 U.S.C. §103(a) over Feridun and Nguyen do not establish a prima facie case of obviousness because the Office has mischaracterized Nguyen as teaching an update consumer that updates a state machine.**

In making out the rejections of the claims, the Office relies on the Nguyen reference as a secondary reference.

Before specifically considering Nguyen, Applicant believes that there are fundamental differences between subject matter disclosed in Applicant’s

disclosure and the subject matter disclosed in Nguyen. Specifically, in various embodiments described and claimed in Applicant's disclosure, implementations that provide a flexible correlation system and/or method that allows a user to correlate multiple events and/or data, are described. One benefit of the implementations is that a non-programmer or application developer can make use of predefined correlation functions that monitor user selected events and/or data parameters.

In one embodiment, a state machine is used to simultaneously monitor different events and data. The state machine may be a class object and may have a number of properties that are modifiable by a non-programmer. For example, the class object may include a string that represents the name of a process being monitored; a string that holds an integer that identifies the number of crashes of the process that will trigger an event if those crashes occur within a particular time period; and a string that holds another integer that defines the particular time period. Therefore, a non-programmer can define such a class object by simply filling-in the values for the process, the number of crashes and the particular time period.

The non-programmer can also create an updating consumer used to update the state of the state machine described in the above-paragraph. In at least some embodiments, the updating consumer is also a class object and is separate from the state machine. In one example, the updating consumer updates an instance of the state machine when a crash occurs. Therefore, the updating consumer ensures that the state machine maintains an up-to-date number of crashes that has been observed by the state machine.

An example updating consumer implementation, from page 19 of Applicant's specification, is illustrated below.

```
Update StateA where ProcessName = ThisEvent.ProcessName  
    set NumCrashes = NumCrashes + 1
```

In this example, the updating consumer updates an instance of state machine StateA, defined above. The ProcessName property is defined as "ThisEvent.ProcessName", which inserts the name of the process that crashed (which is identified in the received crash event) as "ThisEvent". The property NumCrashes is incremented by one each time a crash event is received. Therefore, the updating consumer updates the state machine in response to an identified event.

Nguyen, on the other hand, fails to disclose an updating consumer that updates the state machine. Rather, Nguyen discloses a data evaluation function that simply monitors events and outputs a boolean value when a certain event occurs.

More specifically, Nguyen discloses, at Column 4, Line 66 to Column 5, Line 24, a *data evaluation function that comprises a function that implements a transition condition to evaluate a measured system parameter and, in some instances, information in the status object. The output of the data evaluation function is a boolean value depending on whether the condition is satisfied.* Each state would have at least one data evaluation function to allow transitioning to other states.

Nguyen further instructs that a transition array defines the interconnection of the states. The transition array would comprise an m.times.m array, where m is

the number of states instantiated from the basic state class or the counter state class. The row value of the transition array represents the current state value and the column value indicates the transition state. *Each cell in the transition array includes either 0 or a data evaluation function providing a transition condition.* If the cell (i, j) in row i, column j in the transition array has a zero or null value, then there is no transition from current state i to state j. *If the cell (i, j) has an instance of a data evaluation function or pointer thereto, then that data evaluation function is a transition condition, such that satisfaction of the condition specified in cell (i, j) causes the state machine to move from current state i to transition state j.* Each cell may have a different data evaluation function, as each state is the culmination of a unique set of events or measured parameters causing transitions to such state.

Accordingly, Nguyen uses a data evaluation function that monitors a specific event and will output a boolean value depending on whether the condition is satisfied. A transition array will watch this function, and change states depending on the output boolean value. However, the evaluation function does not physically update the state machine by itself, in response to the identified event.

Applicant has disclosed and claimed an update consumer that is applied to the state machine to update the state machine in response to the identification of an event. In Nguyen, the evaluation function simply outputs a boolean value in response to an event. The transition array then uses this boolean value in order to change states.

Applicant's update consumer acts by itself to update the state machine. On the other hand, Nguyen's evaluation function does not. Therefore, it cannot be

said that the data evaluation function updates the state machine because all that the data evaluation function does is output a boolean value.

Accordingly, the Office has failed to make out a *prima facie* case of obviousness for claims 1-25, 27-31, 34, 35 and 37-43 because Nguyen does not teach or suggest an update consumer, as claimed.

B. The rejections under 35 U.S.C. §103(a) over Feridun and Nguyen do not establish a prima facie case of obviousness because the Office has failed to provide proper motivation for the combination.

In making out a rejection of claims 1-25, 27-31, 34, 35 and 37-43, the Office relies on the combination of Feridun and Nguyen. The Office argues that for each independent claim:

It would be obvious to make the combination since the updating action is invoked when the identified event is determined corresponding to the condition, that the updating action (update consumer) is subscribed to an identified event and is associated with the state machine such that it is applied in response to the identified event. Therefore, it would be obvious to one of ordinary skill in the art to combine the teachings of Feridun with the teachings of Nguyen *in order to provide a greater degree of flexibility in monitoring system states* (citing to Nguyen, Column 1, Lines 58-62).

In making out the rejections, the Office has twisted the language of Nguyen to make it appear as though Nguyen teaches a system that is directed to providing a greater degree of flexibility in monitoring system states. In actuality, column 1, lines 58-62 of Nguyen teaches a “monitoring system that can provide a greater degree of flexibility in monitoring system states *to avoid situations where alarms*

may be excessively generated as measured system parameters continuously fluctuate." (Column 1, lines 58-62).

Feridun has no need whatsoever for a "monitoring system that can provide a greater degree of flexibility in monitoring system states *to avoid situations where alarms may be excessively generated as measured system parameters continuously fluctuate.*" In fact, Feridun teaches that there are five basic rule types. One of these rule types is the "Duplicate Rule." Feridun teaches that "duplicate rules are designed to reduce the event flow traffic. Once a duplicate rule is triggered, it ignores subsequent events of the same type for a specified period of time." (Feridun, column 9, lines 30-32). Accordingly, Feridun does not contemplate a system where it would be important to monitor systems that continuously fluctuate. Instead, Feridun would simply ignore the fluctuations after the first parameter is measured. Accordingly, the motivation given by the Office is misplaced because Feridun has no need to be modified and teaches directly away from a modification to create a "monitoring system that can provide a greater degree of flexibility in monitoring system states *to avoid situations where alarms may be excessively generated as measured system parameters continuously fluctuate.*" Accordingly, one with skill in the art would most assuredly not modify Feridun with the teachings of Nguyen in order to create a "monitoring system that can provide a greater degree of flexibility in monitoring system states *to avoid situations where alarms may be excessively generated as measured system parameters continuously fluctuate.*"

Furthermore, the motivation given by the Office to provide greater flexibility is simply too general. The Office has attempted to make it appear as

though the motivation is proper because the motivation comes directly from Nguyen. However, by truncating the complete sentence from Nguyen, the Office has created a motivation that is too general. One could argue that almost any two references could be combined in order to “provide greater flexibility”.

Accordingly, for this additional reason, the motivation given by the Office is improper.

- C. **The rejections under 35 U.S.C. §103(a) over Feridun and Nguyen do not establish a prima facie case of obviousness because the combination does not teach or suggest all of the claim elements.**

The Claims

Claim 1

Claim 1 recites a computer-implemented method comprising:

- receiving a plurality of events;
- applying the plurality of events to a correlation function, wherein the correlation function is implemented as a state machine and is configured to correlate the plurality of events;
- identifying an event to which an update consumer has subscribed, wherein the update consumer is associated with the state machine;
- applying the update consumer to the state machine in response to the identified event; and
- generating a specific event if the correlation function is satisfied by the plurality of events.

In making out the rejection of this claim, the Office argues that Feridun teaches a computer-implemented method comprising: receiving a plurality of events; applying the plurality of events to a correlation function, wherein the

correlation function is implemented as a state machine and is configured to correlate the plurality of events; and generating a specific event if the correlation function is satisfied by the plurality of events. (citing to column 2, lines 43-62; column 3, lines 9-20; column 9, lines 15-22; column 9, lines 41-57; column 10, lines 5-19; and column 12, lines 32-47). The Office admits that Feridun does not teach or suggest identifying an event to which an *update consumer has subscribed*, wherein the update consumer is associated with the state machine; and *applying the update consumer to the state machine* in response to the identified event. Applicant agrees.

However, on pages 2-3 of the Final Office Action dated December 23, 2005, the Office argues that Nguyen teaches “an object oriented state machine wherein events and parameters are received in the state machine and evaluated to transition conditions to determine if a satisfied condition occurs in order to execute a state action.” (citing to column 2, lines 1-14). The Office further argues that “Nguyen also teaches the state action includes a function of altering the operation of a hardware or software component as well as updating the state of the state machine.” (citing to column 6, lines 26-36; and column 2, lines 35-40). The Office then argues that “since the updating action (update consumer) is invoked when the identified event is determined to corresponding to the condition, that the updating action (update consumer) is subscribed to an identified event and is associated with the state machine such that it is applied in response to the identified event.” Applicant disagrees and submits that the Office has failed to make out a *prima facie* case of obviousness because the combination of Feridun and Nguyen fails to teach all of the elements of claim 1. Specifically, as noted above in section A,

Nguyen does not teach an update consumer. To this extent, Feridun adds nothing of significance. Accordingly, neither Feridun nor Nguyen teaches or suggests identifying an event to which an *update consumer has subscribed*, wherein the update consumer is associated with the state machine; and *applying the update consumer to the state machine* in response to the identified event.

Applicant will now traverse the rejection made by the Office by reproducing and discussing each excerpt from Nguyen cited by the Office as anticipating claim 1.

Column 2, lines 1-14 and 35-40 of Nguyen describe a status object that includes current information related to a state of a monitored system component. Column 2, lines 1-14 of Nguyen is reproduced below for the convenience of the Office.

Nguyen, Column 2, Lines 1-14

A *status object* is provided including *information on a current state* of the monitored system component. There are a plurality of states associated with the monitored system component, wherein *each state is capable of having a state action and at least one transition condition* associated with a transition state. A measured *system parameter is received* and a *determination is made as to whether the received measured system parameter satisfies one transition condition associated with the current state indicated in the status object*. If the received system parameter satisfies one transition condition, *then the state action associated with the transition state associated with the satisfied transition condition is performed*. The current state is set to the transition state in the status object.

As should be apparent from the excerpt above, Nguyen teaches that the status object receives information pertaining to the monitored system component and transitions its state if a transition condition associated with the status object is satisfied. If this occurs, some action is taken by the status object. There is nothing

in this section of the Nguyen patent that teaches or suggests the features from claim 1 that include: “identifying an event to which an update consumer has subscribed, wherein the update consumer is associated with the state machine; [and] applying the update consumer to the state machine in response to the identified event.”

Nguyen further describes the use of a state machine class and a status object class. In order to assist the Office, an excerpt from Nguyen is reproduced below:

Nguyen, Column 2, Lines 35-40

A transition state is associated with each evaluation function. The status object is updated to indicate the transition state as the current state if the associated evaluation function determines that the condition is satisfied.

As should be apparent from the excerpt above, the status object class is updated if a transition state associated with an evaluation function linked to the state machine class is satisfied. Therefore, the status object class can be used to identify a state of the state machine class. This teaching of the Nguyen patent does not approach the features of claim 1 that recite: “identifying an event to which an update consumer has subscribed, wherein the update consumer is associated with the state machine; [and] applying the update consumer to the state machine in response to the identified event.” More specifically, there is nothing in the Nguyen patent that indicates that the status object class is *subscribed* to an event, as recited in claim 1. At most, the status object class in Nguyen monitors a state of the state machine class. This is not the same as being subscribed to an event. Moreover, how can it be fairly said that the status object is *applied* to the state machine in response to the identified event? The Applicant submits that it

cannot be fairly said as such. The status object class in Nguyen simply monitors the state machine class--monitoring is not the same as something being applied.

Column 6, lines 26-35 of Nguyen also fails to teach or suggest the above-discussed features of claim 1. For the convenience of the Office, this excerpt is reproduced below.

Nguyen, Column 6, lines 26-35

action 130: indicates that an action associated with the state is set to occur. If so, then a specified action would be performed. The action may describe the alarm operation to perform if the alarm 128 value is set to true, e.g., send an e-mail, page, etc. Additionally, the action 130 may comprise execution of a program or system command to perform a particular corrective action, such as diagnose a program, close a program or alter the operation of a hardware or software component, etc.

It should be apparent from the excerpt above, that this portion of Nguyen describes the use of an action 130 designed to implement an action when a state of the state machine class changes. This teaching has nothing to do with “identifying an event to which an update consumer has subscribed, wherein the update consumer is associated with the state machine; [and] applying the update consumer to the state machine in response to the identified event”, as recited in claim 1.

All of the excerpts from Nguyen cited by the Office in making out the rejection of claim 1 have been reproduced above. In accordance with the above discussion, Applicant submits that the Nguyen patent does not teach or in any way suggest identifying an event to which an *update consumer has subscribed*, wherein the update consumer is associated with the state machine; and *applying the update consumer to the state machine* in response to the identified event. To

this extent, as admitted by the Office, Feridun adds nothing of significance. Accordingly, for at least this reason the combination of Feridun and Nguyen does not teach or suggest all of the elements of claim 1.

As such, this claim is allowable.

Claims 2-10

Claims 2-10 depend from claim 1 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 1, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claim 11

Claim 11 recites a computer-implemented method comprising:

- receiving a plurality of events;
- receiving a plurality of data elements;
- identifying a plurality of correlation functions configured to correlate the plurality of events and the plurality of data elements, wherein each correlation function is implemented with an associated state machine, and wherein each state machine has an associated update consumer that updates the state of the associated state machine;
- applying the plurality of events and the plurality of data elements to the plurality of correlation functions; and
- generating a specific event if at least one of the plurality of correlation functions is satisfied.

In making out the rejection of claim 11, the Office argues that Feridun teaches a computer-implemented method comprising: receiving a plurality of

events; receiving a plurality of data elements; identifying a plurality of correlation functions configured to correlate the plurality of events and the plurality of data elements, wherein each correlation function is implemented with an associated state machine; applying the plurality of events and the plurality of data elements to the plurality of correlation functions; and generating a specific event if at least one of the plurality of correlation functions is satisfied.

The Office then states that “Feridun fails to teach “applying the update consumer to the state machine in response to the identified event.” Applicant agrees. However, this language is not recited in claim 11. Rather, this language appears in claim 1. Claim 11 recites that each state machine has an *associated update consumer* that *updates the state of the associated state machine*.

In making out the rejection of claim 11, the Office has provided the same argument, with regards to the Nguyen rejection, as was given in making out the rejection of claim 1. Accordingly, for similar reasons to those discussed above, with regards to claim 1, Applicant submits that Nguyen fails to teach that each state machine has an *associated update consumer* that *updates the state of the associated state machine*.

Specifically, as discussed above, column 2, lines 1-14 and 35-40 of Nguyen describe a status object that includes current information related to a state of a monitored system component. The status object receives information pertaining to the monitored system component and transitions its state if a transition condition associated with the status object is satisfied. If this occurs, some action is taken by the status object. (See generally, column 2, lines 1-14.) There is nothing in this section of the Nguyen patent that teaches or suggests the limitation from claim 11

that includes: “each state machine has an associated update consumer that updates the state of the associated state machine.”

The Nguyen patent further describes the use of a state machine class and a status object class. The status object class is updated if a transition state associated with an evaluation function linked to the state machine class is satisfied. Therefore, the status object class can be used to identify a state of the state machine class. (See generally, column 2, lines 24-40.) This teaching of the Nguyen patent does not approach the limitation of claim 11 that recites: “each state machine has an associated update consumer that updates the state of the associated state machine.” More specifically, there is nothing in the Nguyen patent that indicates that the status object class *updates the state* of the state machine class, as recited in claim 11. At most, the status object in Nguyen monitors the state machine class to determine when the state machine class updates. However, that act of monitoring in Nguyen does not teach or suggest that “each state machine has an associated update consumer that updates the state of the associated state machine”, as recited in claim 11.

Column 6, lines 26-35 of Nguyen also fails to teach or suggest the above-discussed limitations of claim 11. This portion of Nguyen describes the use of an action 130 designed to implement an action when a state of the state machine class changes. This teaching has nothing to do with “each state machine has an associated update consumer that updates the state of the associated state machine”, as recited in claim 11.

In accordance with the above, Applicant submits that the Nguyen patent does not teach or in any way suggest that each state machine has an *associated*

update consumer that updates the state of the associated state machine. To this extent, as admitted by the Office, Feridun adds nothing of significance.

Accordingly, for at least this reason the combination of Feridun and Nguyen does not render claim 11 obvious.

For at least this reason, this claim is allowable.

Claims 12-19

Claims 12-19 depend from claim 11 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 11, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claim 20

Claim 20 recites a computer-implemented method comprising:

- identifying a schema for creating state machines, the state machines to correlate at least two events;
- creating an instance of a particular state machine;
- defining transitions for the particular state machine by subscribing to at least one event; and
- applying an update consumer to the particular state machine to update the state of the particular state machine, wherein the update consumer is a class object.

In making out the rejection of this claim, the Office argues that Feridun teaches a computer-implemented method comprising: the state machines to correlate at least two events; creating an instance of a particular state machine; and defining transitions for the particular state machine by subscribing to at least one

event. The Office admits that Feridun does not teach or suggest identifying a schema for creating state machines, and *applying an update consumer* to the particular state machine to *update the state of the particular state machine*, wherein *the update consumer is a class object*. However, the Office argues that Nguyen teaches this subject matter, and that it would have been obvious to combine Feridun with Nguyen.

For the same reasons as discussed above with regards to claim 1 and 11, Applicant submits that Nguyen does not teach or suggest identifying a schema for creating state machines, and *applying an update consumer* to the particular state machine to *update the state of the particular state machine*, wherein *the update consumer is a class object*.

In addition to the arguments raised above, Applicant further submits that Nguyen does not teach the update consumer is a class object. It is unclear to Applicant what the Office actually believes is the updating consumer. However, Nguyen does teach an evaluation function that monitors events and outputs a Boolean value. However, Applicant submits that this evaluation function cannot be said to be an update consumer that updates the state of a particular state machine. All the evaluation function does is output a Boolean value. Further, the evaluation function is a function that outputs a Boolean value. It cannot be said that the evaluation function is a class object.

In accordance with the above, Applicant submits that the Nguyen patent does not teach or in any way suggest applying an update consumer to the particular state machine to update the state of the particular state machine, wherein the update consumer is a class object. To this extent, as admitted by the Office,

Feridun adds nothing of significance. Accordingly, for at least this reason the combination of Feridun and Nguyen does not render claim 20 obvious.

For at least this reason, this claim is allowable.

Claims 21-25 and 27

Claims 21-25 and 27 depend from claim 20 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 20, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claim 28

Claim 28 recites an apparatus comprising:

- a plurality of event consumers; and
- an event correlator coupled to the plurality of event consumers, the event correlator to receive events from at least one event source and to receive data elements from at least one data source, the event correlator further to receive at least one correlation function configured to correlate events and data elements and to apply the received events and the received data elements to the correlation function, wherein the correlation function is implemented by a state machine having an associated update consumer that updates the state of the state machine, and wherein the event correlator generates a specific event if the received events and the received data satisfy the correlation function.

In making out the rejection of this claim, the Office argues that Feridun teaches an apparatus comprising: a plurality of event consumers; and an event correlator coupled to the plurality of event consumers, the event correlator to

receive events from at least one event source and to receive data elements from at least one data source, the event correlator further to receive at least one correlation function configured to correlate events and data elements and to apply the received events and the received data elements to the correlation function, and wherein the event correlator generates a specific event if the received events and the received data satisfy the correlation function. The Office admits that Feridun does not teach or suggest wherein the correlation function is implemented by a state machine having an associated update consumer that updates the state of the state machine. However, the Office argues that Nguyen teaches this subject matter, and that it would have been obvious to combine Feridun with Nguyen.

For the same reasons as discussed above with regards to claims 1 and 11, Applicant submits that Nguyen does not teach or suggest wherein the correlation function is implemented by a state machine having an associated update consumer that updates the state of the state machine. To this extent, as admitted by the Office, Feridun adds nothing of significance. Accordingly, for at least this reason the combination of Feridun and Nguyen does not render claim 28 obvious.

For at least this reason, this claim is allowable.

Claims 29-31 and 34

Claims 29-31 and 34 depend from claim 28 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 28, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claim 35

Claim 35 recites one or more computer-readable media having stored thereon a computer program that, when executed by one or more processors, causes the one or more processors to:

- receive a plurality of events;
- identify a plurality of correlation functions configured to correlate the plurality of events, wherein each of the plurality of correlation functions is implemented as a state machine having an associated update consumer;
- apply the plurality of events to the plurality of correlation functions to determine whether any of the plurality of correlation functions are satisfied by the plurality of events, wherein the plurality of events are applied by causing update consumers associated with each event to update the state of the associated state machine; and
- generate a specific event if one of the plurality of correlation functions is satisfied by the plurality of events.

In making out the rejection of this claim, the Office argues that Feridun teaches one or more computer-readable media having stored thereon a computer program that, when executed by one or more processors, causes the one or more processors to: receive a plurality of events; identify a plurality of correlation functions configured to correlate the plurality of events, wherein each of the plurality of correlation functions is implemented as a state machine; apply the plurality of events to the plurality of correlation functions to determine whether any of the plurality of correlation functions are satisfied by the plurality of events; and generate a specific event if one of the plurality of correlation functions is satisfied by the plurality of events.

The Office admits that Feridun does not teach or suggest wherein the plurality of events are applied by causing update consumers associated with each event to update the state of the associated state machine. However, the Office argues that Nguyen teaches this subject matter, and that it would have been obvious to combine Feridun with Nguyen.

For the same reasons as discussed above with regards to claims 1 and 11, Applicant submits that Nguyen does not teach or suggest wherein the plurality of events are applied by causing update consumers associated with each event to update the state of the associated state machine. To this extent, as admitted by the Office, Feridun adds nothing of significance. Accordingly, for at least this reason the combination of Feridun and Nguyen does not render claim 35 obvious.

For at least this reason, this claim is allowable.

Claims 37-39

Claims 37-39 depend from claim 35 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 35, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claim 40

Claim 40 recites a computer-implemented method comprising:

- receiving events from event providers;
- creating a first state machine;

- creating a second state machine;
- associating a first event type with the first state machine, wherein the first state machine has an associated first update consumer to update the state of the first state machine;
- associating a second event type with the second state machine, wherein the second state machine has an associated second update consumer to update the state of the second state machine;
- in response to receiving an event having a first event type, applying the first update consumer to the first state machine;
- in response to receiving an event having a second event type, applying the second update consumer to the second state machine; and
- if the events are correlated:
 - generating an additional event; and
 - sending the additional event to an event consumer.

In making out the rejection of this claim, the Office argues that Feridun teaches a computer-implemented method comprising: receiving events from event providers; creating a first state machine; creating a second state machine; associating a first event type with the first state machine, associating a second event type with the second state machine, in response to receiving an event having a first event type, performing an action to the first state machine, in response to receiving an event having a second event type, applying an action to the second state machine; and if the events are correlated: generating an additional event; and sending the additional event to an event consumer.

The Office admits that Feridun does not teach or suggest wherein the first state machine has an associated first update consumer to update the state of the first state machine; and wherein the second state machine has an associated second update consumer to update the state of the second state machine; and in response to receiving an event having a first event type, applying the first update consumer

to the first state machine; and in response to receiving an event having a second event type, applying the second update consumer to the second state machine.

However, the Office argues that Nguyen teaches this subject matter, and that it would have been obvious to combine Feridun with Nguyen.

For the same reasons as discussed above with regards to claims 1 and 11, Applicant submits that Nguyen does not teach or suggest wherein the first state machine has an associated first update consumer to update the state of the first state machine; and wherein the second state machine has an associated second update consumer to update the state of the second state machine; and in response to receiving an event having a first event type, applying the first update consumer to the first state machine; and in response to receiving an event having a second event type, applying the second update consumer to the second state machine. To this extent, as admitted by the Office, Feridun adds nothing of significance. Accordingly, for at least this reason the combination of Feridun and Nguyen does not render claim 40 obvious.

For at least this reason, this claim is allowable.

Claims 41-43

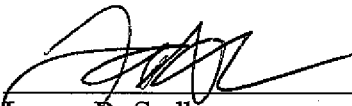
Claims 41-43 depend from claim 40 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 140 are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Conclusion

The Office has not established a prima facie case of obviousness.
Accordingly, Applicant respectfully requests that the rejections be overturned and
that the pending claims be allowed to issue.

Respectfully Submitted,

Dated: 9/22/06

By: 
Lance R. Sadler
Lee & Hayes, PLLC
Reg. No. 38,605
(509) 324-9256 ext. 226

(8) Appendix of Appealed Claims

1. (Previously Presented) A computer-implemented method comprising:
receiving a plurality of events;
applying the plurality of events to a correlation function, wherein the correlation function is implemented as a state machine and is configured to correlate the plurality of events;
identifying an event to which an update consumer has subscribed, wherein the update consumer is associated with the state machine;
applying the update consumer to the state machine in response to the identified event; and
generating a specific event if the correlation function is satisfied by the plurality of events.

2. (Previously Presented) A method as recited in claim 1 wherein the update consumer is a class object.

3. (Original) A method as recited in claim 1 further including:
receiving a data element; and
applying the data element and at least one of the plurality of events to the correlation function.

4. (Original) A method as recited in claim 1 further including:
receiving a plurality of data elements; and
applying the plurality of data elements and the plurality of events to the correlation function.

5. (Original) A method as recited in claim 1 further including communicating the specific event to at least one event consumer that subscribed to the specific event.

6. (Original) A method as recited in claim 1 further including continuing to receive additional events and apply the additional events to the correlation function if the correlation function is not satisfied by the plurality of events.

7. (Original) A method as recited in claim 1 further including resetting the correlation function after generating a specific event.

8. (Original) A method as recited in claim 1 further including:
creating an instance of a particular state machine; and
defining transitions for the particular state machine by subscribing to at least one event.

9. (Previously Presented) A method as recited in claim 8 further including deleting the instance of the particular state machine if the instance of the particular state machine reaches a final state.

10. (Original) One or more computer-readable memories containing a computer program that is executable by a processor to perform the method recited in claim 1.

11. (Previously Presented) A computer-implemented method comprising:

receiving a plurality of events;

receiving a plurality of data elements;

identifying a plurality of correlation functions configured to correlate the plurality of events and the plurality of data elements, wherein each correlation function is implemented with an associated state machine, and wherein each state machine has an associated update consumer that updates the state of the associated state machine;

applying the plurality of events and the plurality of data elements to the plurality of correlation functions; and

generating a specific event if at least one of the plurality of correlation functions is satisfied.

12. (Previously Presented) A method as recited in claim 11 further comprising deleting a particular state machine when the particular state machine reaches a final state.

13. (Previously Presented) A method as recited in claim 11 wherein each update consumer is a class object.

14. (Original) A method as recited in claim 11 further including communicating the specific event to at least one event consumer that subscribed to receive the specific event.

15. (Original) A method as recited in claim 11 further including:
receiving additional events;
receiving additional data elements; and
applying the plurality of events, the additional events, the plurality of data elements and the additional data elements to the plurality of correlation functions.

16. (Previously Presented) A method as recited in claim 11 further including:
receiving additional events;
receiving additional data elements;
receiving additional correlation functions; and
applying the plurality of events, the additional events, the plurality of data elements and the additional data elements to the plurality of correlation functions and the additional correlation functions.

17. (Original) A method as recited in claim 16 further including generating the specific event if at least one of the plurality of correlation functions or at least one of the additional correlation functions is satisfied.

18. (Original) A method as recited in claim 11 wherein the specific event generated is dependent on which correlation function is satisfied.

19. (Original) One or more computer-readable memories containing a computer program that is executable by a processor to perform the method recited in claim 11.

20. (Previously Presented) A computer-implemented method comprising:
identifying a schema for creating state machines, the state machines to correlate at least two events;

creating an instance of a particular state machine;

defining transitions for the particular state machine by subscribing to at least one event; and

applying an update consumer to the particular state machine to update the state of the particular state machine, wherein the update consumer is a class object.

21. (Original) A method as recited in claim 20 further including deleting the particular state machine if the particular state machine reaches a final state.

22. (Original) A method as recited in claim 20 wherein the particular state machine includes a timer, the method further including deleting the particular state machine if the timer expires.

23. (Original) A method as recited in claim 20 wherein the particular state machine correlates at least one event and at least one data element.

24. (Original) A method as recited in claim 20 wherein the particular state machine correlates a plurality of events and a plurality of data elements.

25. (Original) A method as recited in claim 20 further including determining a current state of the particular state machine.

26. (Canceled).

27. (Original) One or more computer-readable memories containing a computer program that is executable by a processor to perform the method recited in claim 20.

28. (Previously Presented) An apparatus comprising:

a plurality of event consumers; and

an event correlator coupled to the plurality of event consumers, the event correlator to receive events from at least one event source and to receive data elements from at least one data source, the event correlator further to receive at least one correlation function configured to correlate events and data elements and to apply the received events and the received data elements to the correlation function, wherein the correlation function is implemented by a state machine having an associated update consumer that updates the state of the state machine, and wherein the event correlator generates a specific event if the received events and the received data satisfy the correlation function.

29. (Original) An apparatus as recited in claim 28 wherein the event correlator communicates the specific event to the plurality of event consumers.

30. (Original) An apparatus as recited in claim 28 wherein the event correlator communicates the specific event to event consumers that have requested to receive the specific event.

31. (Original) An apparatus as recited in claim 28 wherein the event correlator communicates the specific event to a plurality of filters, wherein each of the plurality of filters is associated with one of the plurality of event consumers.

32-33. (Canceled).

34. (Original) An apparatus as recited in claim 28 wherein the event correlator continues to receive additional events and additional data elements and apply the additional events and the additional data elements to the correlation function.

35. (Previously Presented) One or more computer-readable media having stored thereon a computer program that, when executed by one or more processors, causes the one or more processors to:

receive a plurality of events;

identify a plurality of correlation functions configured to correlate the plurality of events, wherein each of the plurality of correlation functions is implemented as a state machine having an associated update consumer;

apply the plurality of events to the plurality of correlation functions to determine whether any of the plurality of correlation functions are satisfied by the plurality of events, wherein the plurality of events are applied by causing update consumers associated with each event to update the state of the associated state machine; and

generate a specific event if one of the plurality of correlation functions is satisfied by the plurality of events.

36. (Canceled).

37. (Previously Presented) One or more computer-readable media as recited in claim 35 wherein each state machine is a class object.

38. (Original) One or more computer-readable media as recited in claim 37 further causing the one or more processors to identify a current state of the state machine.

39. (Original) One or more computer-readable media as recited in claim 35 further causing the one or more processors to:

create a new instance of a state machine to implement a particular correlation function; and

define transitions for the new instance of the state machine by subscribing to at least one event.

40. (Previously Presented) A computer-implemented method comprising:

receiving events from event providers;

creating a first state machine;

creating a second state machine;

associating a first event type with the first state machine, wherein the first state machine has an associated first update consumer to update the state of the first state machine;

associating a second event type with the second state machine, wherein the second state machine has an associated second update consumer to update the state of the second state machine;

in response to receiving an event having a first event type, applying the first update consumer to the first state machine;

in response to receiving an event having a second event type, applying the second update consumer to the second state machine; and

if the events are correlated:

generating an additional event; and

sending the additional event to an event consumer.

41. (Previously Presented) The method as recited in claim 40, further comprising deleting the first state machine if the first state machine reaches a final state.

42. (Previously Presented) The method as recited in claim 40, wherein the additional event is sent to the event consumer through a filter associated with the event consumer.

43. (Previously Presented) The method as recited in claim 40, wherein the event consumer includes at least one of an event logging consumer, an event forwarding consumer, a mail consumer, and a scripting consumer.

(9) Evidence appendix. None

(10) Related proceedings appendix. None